#3

Europäisches Patentamt | European Patent Office | Office européen des brevets

Bescheinigung | Certificate | Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr. | Patent application No. | Demande de brevet n°

99480088.6

CERTIFIED COPY OF PRIORITY DOCUMENT

Der Präsident des Europäischen Patentamts:
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

I.L.C. HATTEN-HECKMAN

DEN HAAG, DEN
THE HAGUE, 21/10/99
LA HAYE, LE

Europäisches Patentamt | European Patent Office | Office européen des brevets

Blatt 2 der Bescheinigung
Sheet 2 of the certificate
Page 2 de l'attestation

Anmeldung Nr.:
Application no.: 99480088.6
Demande n°:

Anmeldetag:
Date of filing: 23/09/99
Date de dépôt:

Anmelder:
Applicant(s):
Demandeur(s):
INTERNATIONAL BUSINESS MACHINES CORPORATION
Armonk, NY 10504
UNITED STATES OF AMERICA

Bezeichnung der Erfindung:
Title of the invention:
Titre de l'invention:
System and method for improving gateways transparency

In Anspruch genommene Priorität(en) / Priority(ies) claimed / Priorité(s) revendiquée(s)

Staat: State: Pays:	Tag: Date: Date:	Aktenzeichen: File no. Numéro de dépôt:

Internationale Patentklassifikation:
International Patent classification:
Classification internationale des brevets:
/

Am Anmeldetag benannte Vertragstaaten:
Contracting states designated at date of filing: AT/BE/CH/CY/DE/DK/ES/FI/FR/GB/GR/IE/IT/LI/LU/MC/NL/PT/SE
Etats contractants désignés lors du depôt:

Bemerkungen:
Remarks:
Remarques:

SYSTEM AND METHOD FOR IMPROVING GATEWAYS TRANSPARENCY.

Field of the Invention

The present invention relates to the Internet and more particularly applies to gateways and proxies used by Internet Service Providers (ISPs) and enterprise networks administrators at the boundary of their networks.

Background of the Invention

The Internet is actually a worldwide IP network that links many different organizations. The Internet is not a centralized organization but a collection of different networks from various sources, governmental, educational and commercial. Internet routing is done by many Internet providers, government departments and private service companies who establish connections among themselves and build the base of the network. Organizations and individuals connected to the Internet are usually bound to one provider and so may communicate with any other connected organization and individual across the inter-provider routes that are made of expensive communications lines often referred to as 'peer lines'. To cope with the explosion of the Internet over the past years a rapid expansion in bandwidth and other resources deployed by the ISPs was required. Then, to contain their operational costs, ISPs have largely used proxy caching which can significantly reduce bandwidth costs by retaining locally highly used information rather than accessing it from a remote server, through an expensive link, each time it is requested by an end-user (ISP's customers and users). The caching proxy function is also beneficial to the end-user who may thus enjoy good response time. The function is carried out by a proxy

server, a Web server, which takes over the responsibility of retrieving Internet data for multiple browser clients. Client requests are sent to the servers through the proxy. In other words, the client has to be configured to send its request to the proxy first, and then it is the proxy that forwards the client's request to the server, acting on behalf of the originating client. The remote Web server does not even see the IP address of the client in the packet headers, but only the IP address of the proxy server. Once the proxy receives the information from the server, it forwards the information to the requesting client. This way the proxy function can be used to provide address security and optionally, through specific proxy features, to support additional functions, such as request filtering or modification that the service provider may want to implement.

Thus, a traditional proxy server receive requests for URLs (Uniform Resource Locator) from clients and then forward them to the destination Web server. Hence, those of the retrieved Web documents, that are considered to be cacheable according to the Hypertext Transport Protocol (HTTP), are saved. Proxy server can then serve subsequent requests for cached documents from its local cache. Clients get the information faster and network bandwidth utilization is reduced.

Although the proxying technique is advantageous both for the Internet provider, which can somehow limit its bandwidth requirement on peer lines (while the number of Internet users is exponentially growing), and for the clients who get a better response time, it has created problems of two kinds. Firstly, as mentioned here above, the origin IP address of the client is lost in the packet headers received by the servers since the proxy acts as a relay between them. Thus, the trace ability of the exchanges is impaired. This may become a serious problem if a wrong doer, a hacker, is attempting to attack a site or tries to disseminate a virus. In which case the Web site or the end-adressee of a mail, which realizes it

has been subject to an attack, can only be aware of the proxy address as the origin of the malicious IP packets. This may not help a lot if the ISP from which packets have been originated is hosting thousands and sometimes tens or hundreds of thousands of clients for the biggest. Secondly, having a proxy assumes that the client browsers are personalized for that proxy, the users become proxy-aware, which poses serious scale ability problems when a successful provider wants to grow which, in this business, means growth number in the range of 10% a month. Configuring and re-configuring the end-user browsers can become a cumbersome and costly task that may have severe adverse commercial impacts and anyway, contributes significantly to increase the administrative cost of managing a network.

As a consequence transparent proxying was introduced. This technique implicitly assumes that there is a single gateway (or at least a limited number of them) through which all the clients connected to an ISP network or all the users on an intranet are bound to get through to access to the Internet. In practice this assumption always hold. For instance, proxy caches, here above discussed, need to be placed at gateways to be efficient and other considerations like security tend to limit to a single point the access of a sub-network so as it is convenient to watch all in and out traffic. Then, transparent proxying manages to redirect all client sessions passing the gateway to local proxy servers in a fully transparent way. Clients (both users and software i.e., client browsers) do not know their session is handed over to a proxy process: they still think they have a direct connection with the target they specified. To achieve this, transparent proxying relies on port numbers hence, it only works for TCP (Transport Control Protocol) and UDP (User Datagram Protocol) used by higher-layer protocols of the IP suite of protocols such as HTTP i.e., the World Wide Web or simply the Web and the Domain Name Service (DNS) protocol respectively. Conceptually, TCP and UDP include also, on top

of the IP destination and source addresses of a datagram, a protocol port number, allowing the sender to distinguish among multiple applications programs on the remote machine. Because there are "well-known port numbers", a list of which can be found in RFC1700 (i.e., a Request For Comment of the Internet Engineering Task Force or IETF) and "privilege ports" (i.e. port numbers below 1023), a router acting as the gateway of a sub-network connected on the Internet can be programmed to intercept e.g. all HTTP requests on port 80, the port number for the applications using this IP protocol. Then, all HTTP requests may be indeed forwarded transparently to a proxy server as requested without having to personalize client browers. A discussion on this and more can be found in a publication by the International Technical Support Organization of IBM Corporation, P.O. Box 12195, Research Triangle Park, NC 27709 U.S.A, under the title *'Web Caching and Filtering with IBM Websphere Performance Pack'*, dated March 1999.

Although the here above scheme works well and is largely used it can become the source of many problems. If a service normally uses a well-known port, that does not mean that it can not use another port. This must be considered because it might be used to circumvent the gateways restrictions either by an outsider or an insider if, as it is often the case, on top of being just a caching proxy it implements logging, filtering and security functions. Often, weaknesses are not directly created by outsiders, but by unhappy insiders who consider the gateway to be unnecessarily restrictive. An insider that wants to provide an outside access that is not permitted may use a nonstandard port in order to do it. For example, if one prevents users from providing HTTP servers but allow connections from outside to non privileged ports (i.e. equal or greater than 1023), a user can provide HTTP access using a port other than 80 thus, escaping the transparent proxy server and its logging, filtering and security functions. Also an outside privileged port might be used by an outsider to circumvent the gateway. If, for example, it is

allowed from outside to access from TCP port 20 (a port usually used by a File Transfer Protocol or FTP server for data transfer), an outsider may use this port in order to run another service, for example, a Telnet client. Because Telnet is the protocol used to emulate terminal sessions from within the network, like insiders, this may have devastating consequences. Transparent proxying is further illustrated as prior art in figure 1.

Another popular approach to implement network gateways uses a proxy server running a networking proxy protocol referred to as SOCKS. This technique enables hosts on one side of the proxy server (e.g., clients) to gain full access to hosts (e.g., servers) on the other side of the proxy server without requiring direct IP reach ability. However, SOCKS not solely require that protocol be run in the proxy server itself it also assumes that each client is personalized i.e., 'socksified' so as to become able to interact with the proxy server. SOCKS, from which is derived the present invention, is further discussed as prior art in figure 2.

Thus, it is an object of the invention to overcome the shortcomings, as noted above, of the prior art yet retaining all the advantages of using a transparent proxy function which does not require that end-user or client be personalized whatsoever.

It is a further object of the invention not to bind the transparency of a proxy function to the examination of the TCP port from which a service is usually carried out.

Further advantages of the present invention will become apparent to the ones skilled in the art upon examination of the drawings and detailed description. It is intended that any additional advantages be incorporated herein.

Summary of the Invention

In a client-server environment, a method and system are disclosed for granting transparency to the compulsory gateway of an IP network versus one or more client applications run by one or more end-users on one or more machines connected on the IP network. When client applications have to access, on request of end-users, server applications beyond said compulsory gateway, following is performed:

- Upon receiving, in the compulsory gateway, a request to access one server application from one client application on request of one end-user:
  - A directory, comprising inputs for every end-user of the IP network, is interrogated.
  - Parameters, associated to the end-user having issued the request from a client application, are retrieved.
  - An access to the application server, on behalf of the client application in accordance to the retrieved parameters for the end-user, is then attempted.
- Upon having successfully established a link to the application server on behalf of the client application for the end-user:
  - Data between the client application and the application server are relayed.
- Or, upon having failed to establish a link to the application server:
  - End-user of the client application is informed that server application is unavailable.

The method and system of the invention permit that client applications need not to be personalized (they do not have to be gateway-aware) to become capable of accessing external resources located beyond the gateway of the IP network they are connected to. Invention permits that gateway acts on behalf of the end-users to access remote server applications through a client agent retrieving end-user parameters from a directory having entries for all end-users of the IP network. This greatly eases the task of managing an IP network and guarantees automatically that all end-users (since the gateway is becoming transparent to them) are using the facilities network manager may want to put in place to improve, for example, the security and performance of its network while skiping the burden of having to reconfigure every client application and end-user.

Brief Description of the Drawings

Figure 1 Describes one form of prior art referred to as transparent proxy.

Figure 2 Depicts another form of prior art known as SOCKS.

Figure 3 describes transparent SOCKs per the invention.

Detailed Description of the Preferred Embodiment

Figure 1 illustrates one form of prior art i.e., transparent proxying, often used to implement an application cache in a proxy server [100], possibly having access to ample storage facilities [101], and which prevents from having to configure each client browser [110]. Transparent proxying function is carried out from a router [120] which is programmed to divert packets destined to a specified port (e.g., port 80 for HTTP) to the proxy server [100]. The clients are configured so that all the packets they send, that are destined for the Internet [140], must pass through the diverting router [120] or the clients network [130] is such that router is a choke point through which all outside traffic is anyway bound to go through. Then, router sends all packets destined for port 80 to proxy [100]. This latter intercepts the requests and processes them as usual in a cache; that is, if the content is in the cache, then it just sends the content to the client [110]; otherwise, it retrieves the content from the Web server [150] on the Internet and then sends the content to the client possibly retaining a copy locally for future use. Thanks to the transparent proxying technique client is never aware that a proxy server is being used and because it needs not to be specially configured to take advantage of it this a guarantee for the network administrator that, on one hand, all clients benefit, on the average, of a good response time while, on the other hand the network bandwidth utilization over expensive lines [145] used to connect to the Internet is reduced. However, this technique is completely relying on the port number used by protocol to work properly. In practice numerous escapes are possible which can become the source of many problems especially when security is considered.

Figure 2 is discussing another sort of prior art to the present invention referred to as Sockets Server or just as SOCKS in the literature on IP networks. In this case the gateway of the ISP or Enterprise network [200] is a proxy server [220] i.e., any computer-like machine or work station capable of running the TCP/IP suite of protocols or a subset of it. Then, SOCKS is a proxy protocol [221] run at the application level on a proxy server. From the application server's perspective [252], the proxy server is becoming the client. When an end client [201] wants to make a connection to an application running on a server [251] through the Internet [250], the client connects to the proxy server [220]. The application server's address and port number are passed to the proxy server via a proxy protocol. The proxy server then connects to the application server [252]. Once the connection [240] to the application server is established, the proxy server relays data between the client and the applications server.

Currently, there are two versions of the SOCKS protocol, version 4 and version 5. The SOCKS version 4 protocol is referred to as "SOCKS V4". Similarly, the SOCKS version 5 protocol is referred to as "SOCKS V5" whose specifications are laid out in RFCs i.e., Request For Comments of the Internet Engineering Task Force or IETF, 1928 (SOCKS Protocol Version 5) and 1929 (Username/Password Authentication for SOCKS V5).

Because of its simplicity and flexibility, SOCKS has been widely used providing for transparent network access across firewalls, easy deployment of authentication/encryption methods, rapid deployment of new network applications, simple extension of network security policy and flexible network traffic screening/filtering. However, all these advantages are obtained at the expense of a complication on the client side. This is due to the modifications required on the protocol stack [202] of the client machines [201]. The protocol stack, on each client machine, must be 'socksified' so as to

be able to interact with the SOCKS [221] of the proxy server [220] in order to carry out, in proxy server, functions already mentioned above such as authentication, filtering and address translation. Moreover, the client is also required to configure SOCKS server address and location so that the socksified stack and the application on top of it will be directed to the socks server prior to be relayed to the application server. In practice SOCKS includes two primary components, the SOCKS server i.e., the piece of software running at the application level [221] on a proxy server and the SOCKS client library, i.e., a software between the client's application and transport layers in the client machine [203].

Thus, despite of all its advantages, SOCKS fails meeting an important concern of network administrators which is that, ideally, the end user should not be affected whatsoever by the solutions adopted to administrate and run a network.

Figure 3 illustrates the principles of the invention. Figure 3(a) is, for the sake of clarity, the control flow of the here above prior art to the invention i.e., SOCKS Versions 4 and 5. The chief difference between the two versions is that Version 5 of SOCKS adds authentication [300] to version 4 [305]. The invention, illustrated in figure 3(b), is based on the idea that instead of performing the socks protocol between client [310] and socks server [320] which requires that each client be 'socksified', an agent [315], at the SOCKS server, plays client's role on behalf of the client. Then, processes like authentication [325] takes place between this agent and a subscriber directory or database (e.g., [361]) where policies and generally all parameters specific to clients were previously stored. Hence, an initial connection request [345] from a client, captured by the transparent socks manager [350], triggers an interrogation of the directory [346] so that to determine first what version of SOCKS client uses. Depending upon the result of this first interrogation client request is

directed to SOCKS V5 agent [335] or SOCKS V4 agent [340]. When SOCKS V5 agent is selected directory is interrogated again [356] [361] to find what methods [355] are used by client and what kind of authentication parameters [360] are set. Then, (this is however the first step if SOCKS V4 was selected earlier), client agent [315] passes the request to the socks server [320] which starts processing the connection request [370]. At this point leading part of client's application data [375] may already have been obtained so that it can be thoroughly examined. This step, although optional, and which can be carried out at various levels of sophistication, open the door to many possibilities that were not possible with standard SOCKS like implementing a proxy cache for a certain type of applications such as HTTP already described in figure 1 with transparent proxying. The leading part of the application data, which contains the headers of the protocols in use, can thus be examined and parsed so that to retrieve through a further interrogation of the directory [376] all information necessary to better process the application data and application protocols used. One example being that transparent SOCKS may thus determine what server (local, remote or none if request cannot be honored) is best suited, when several possibilities exist, to serve client requests and to keep using it consistently while client session is on. Then, under normal circumstances, SOCKS server establishes the connection with the application server and sends a circuit status to the SOCKS agent (which has however the freedom of resetting the connection with the client if something unexpected occurs on application server side). Finally, the socks server establishes the data relay [380] between the application server and the client. This latter ignores it is actually dealing with a transparent SOCKS and has not to be configure.

However, it is worth to reemphasize that the scheme of the invention works under the assumption (which always holds in practice) that transparent SOCKS is installed at the gateway of a network which must be a choke point for all in

and out traffic like with previous art especially the transparent proxy described in figure 1. Thus, connection requests [345] issued by the clients towards the application server (regardless of the destination IP address) all reached transparent SOCKS and are accepted by it. During the last interrogation [376] of the directory final destination to best serve the request is determined. On the way from the socks server to the application server, the socks server may use the IP address of the client, on behalf of the client, to represent the client vis-a-vis the application server, transparently. Because the socks server logically bind two TCP connections (client to socks and socks to application server), and since it is on the path between client and application server, it can determine at any time which flow on one side is associated with which flow on the other side keeping all transactions between clients and servers always consistent.

Therefore, the scheme of the invention allows a complete transparency when it is convenient to do so. If we consider again figure 2 in which proxy server would now run the transparent SOCKS per the invention and if network [200] is the one of an ISP serving individual customers this latter may decide to open connections like [240] by inserting as the source address the actual IP address of the originator of the request (and not the IP address of the gateway [220]) so as, in case of problem the end application targeted by a client of the ISP network [200] may better pinpoint the true origin of the request beyond network gateway [220]. Still, the opposite is possible simultaneously. In which case, transparent socks, may be programmed such as it hides internal IP addresses for some protocols, or a subset of clients using a certain application protocol, in order to keep their privacy or e.g., for security considerations.

Thus, the invention allows to be completely flexible as far as the way client requests from an ISP/Enterprise network are processed and does not require whatsoever, that clients

become gateway-aware which guarantees that the solutions put in place by network administrators to improve their networks (response times, costs etc.) are actually effective for all end clients.

Finally, those skilled in the art will recognize that the invention does not require anything but well know solutions to allow implementation. Especially, building and managing a directory of end clients in the transparent SOCKS, containing all the permanent or dynamic information (e.g., the temporary IP address assigned by ISP to their clients for the duration of a session) necessary to carry out the invention, requires only standard techniques and products. Directory may be organized in many different ways while remaining within the scope of the invention.

Claims:

1. In a client-server environment, a method for granting transparency to the compulsory gateway of an IP network versus one or more client applications run by one or more end-users on one or more machines connected on said IP network, said client applications having to access, on request of said end-users, server applications beyond said compulsory gateway, said method comprising the steps:

upon receiving, in said compulsory gateway, a request to access, from one of said client applications on request of one said end-users, one of said server applications:

interrogating a directory comprising inputs for every end-user of said IP network; and

retrieving parameters associated to said end-user for said request from said client application; and

attempting to access said application server on behalf of said client application in accordance to said retrieved parameters for said end-user; and

upon having successfully established a link to said application server on behalf of said client application for said end-user:

relaying data between said client application and said application server; or

upon having failed to establish a link to said application server:

informing said end-user of said client application that server application is unavailable.

2. The method according to claim 1 including the preliminary step of:

creating, in said compulsory gateway of said IP network, a directory including entries for every end-user on said IP network.

3. The method according to any one of the preceding claims including the further step of:

updating, in said compulsory gateway of said network, the directory of said end-users, said step of updating the directory further including the steps of:

deleting or disabling entries for those of said end-users that disconnect;

adding or enabling entries for those of said end-users that connect;

updating said entries of said end-users comprising dynamic or temporary parameters whenever said parameters are changing while connected.

4. The method according to any one of the preceding claims wherein the step of retrieving parameters associated to said end-user for said request from said client application further includes the steps of:

obtaining in said compulsory gateway leading data from said client application having issued said request for said end-user;

parsing said leading data;

determining which protocol said client application is currently using;

interrogating said directory at said end-user entry;

retrieving further parameters associated to said protocol;

acting on in accordance with said protocol for said further retrieved parameters.

5. The method according to any one of the previous claims wherein said server applications beyond said compulsory gateway are accessed through various networks including:

the Internet;

a public IP network;

a private IP network;

a combination of.

6. A system, in particular a gateway of an IP network, comprising means adapted for carrying out the method according to any one of the previous claims.

7. A computer-like readable medium comprising instructions for carrying out the method according to any one of the claims 1 to 5.

SYSTEM AND METHOD FOR IMPROVING GATEWAYS TRANSPARENCY.

Abstract

In a client-server environment, a method and system are disclosed for granting transparency to the compulsory gateway of an IP network versus client applications run by the end-users connected on the IP network. When client applications have to access, on request of the end-users, server applications beyond the compulsory gateway this latter acts on behalf of them to access the requested remote resources eventually, relaying data between client and server applications. This is achieved without requiring any personalization on client end thus, guarantees that since gateway is becoming transparent to client applications all improvements and modifications brought to the IP network actually fully benefit to all end-users.

Figure 3(b).

1/4
Lamberton et al.

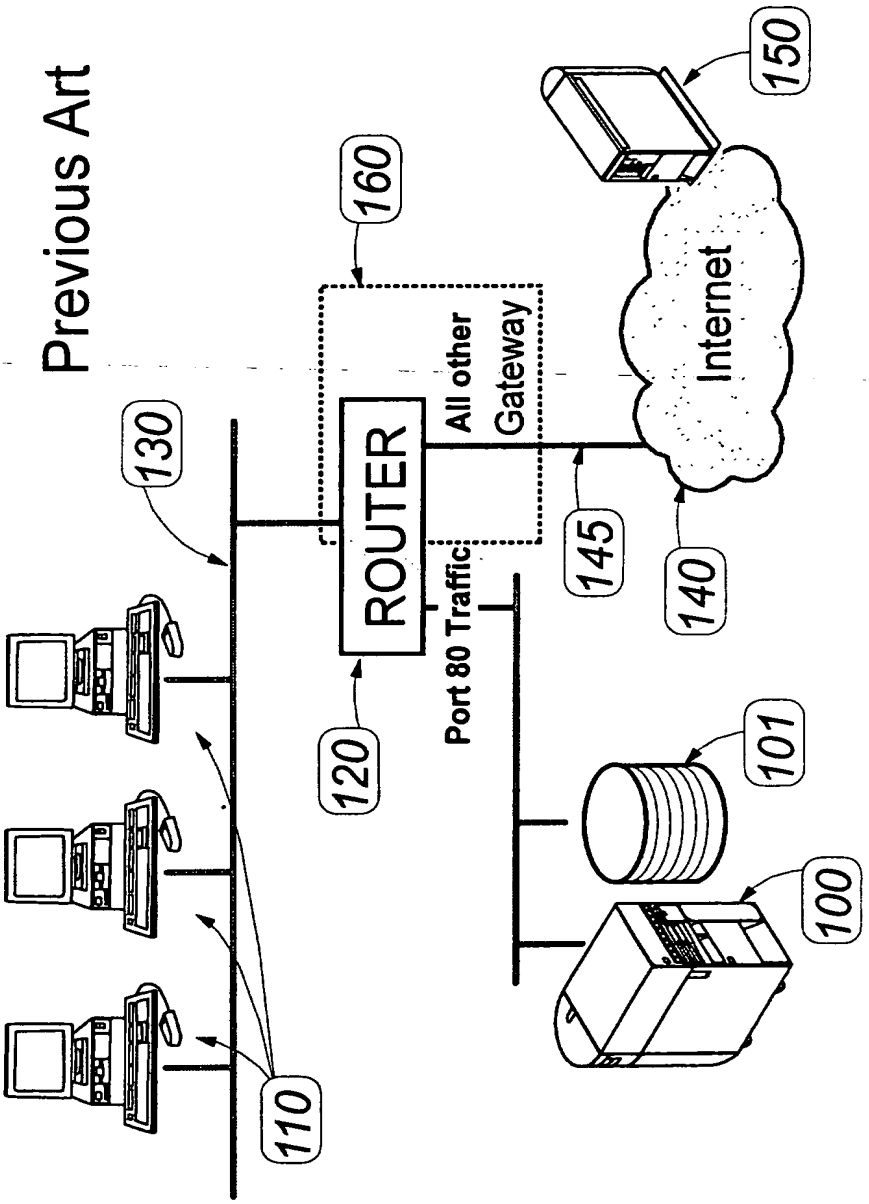

Figure 1

FR 9 99 061

2/4
Lamberton et al.

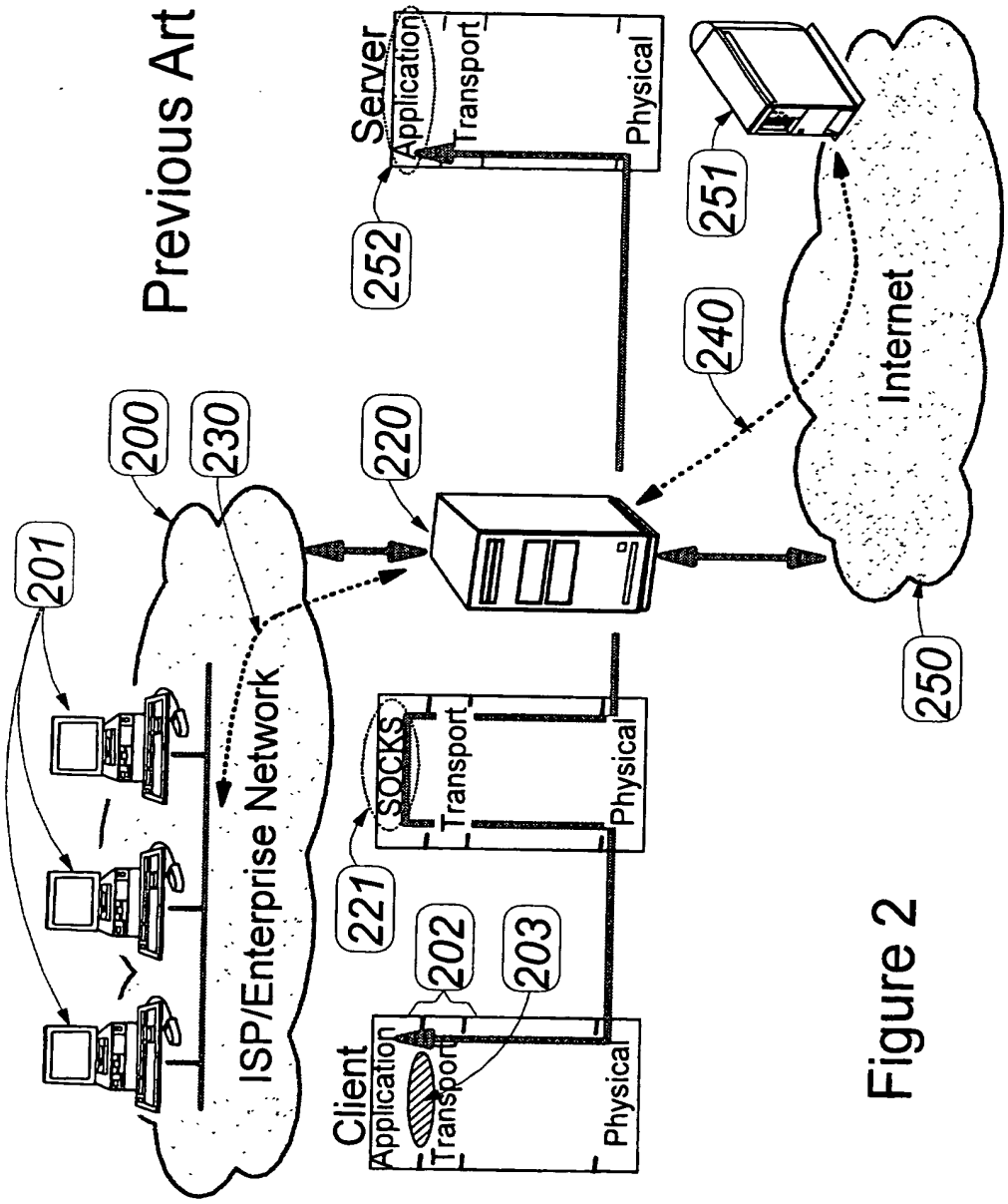

Figure 2

FR 9 99 061

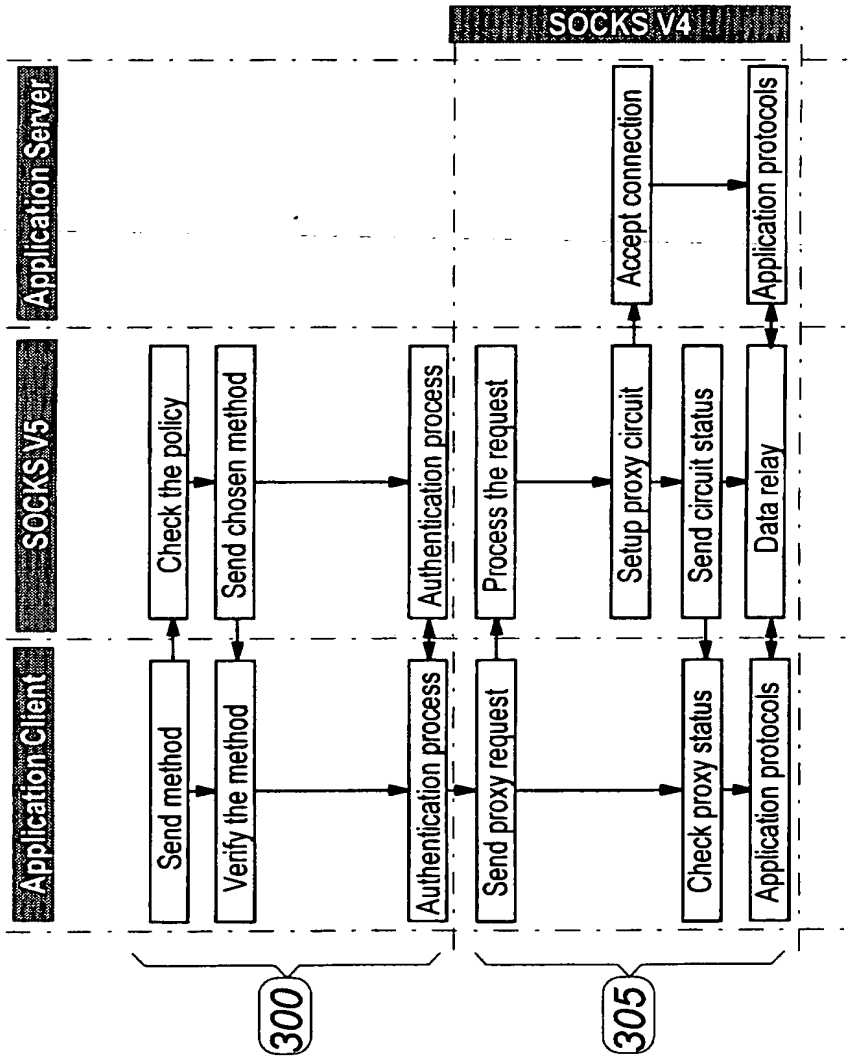

Figure 3(a)

FR 9 99 061

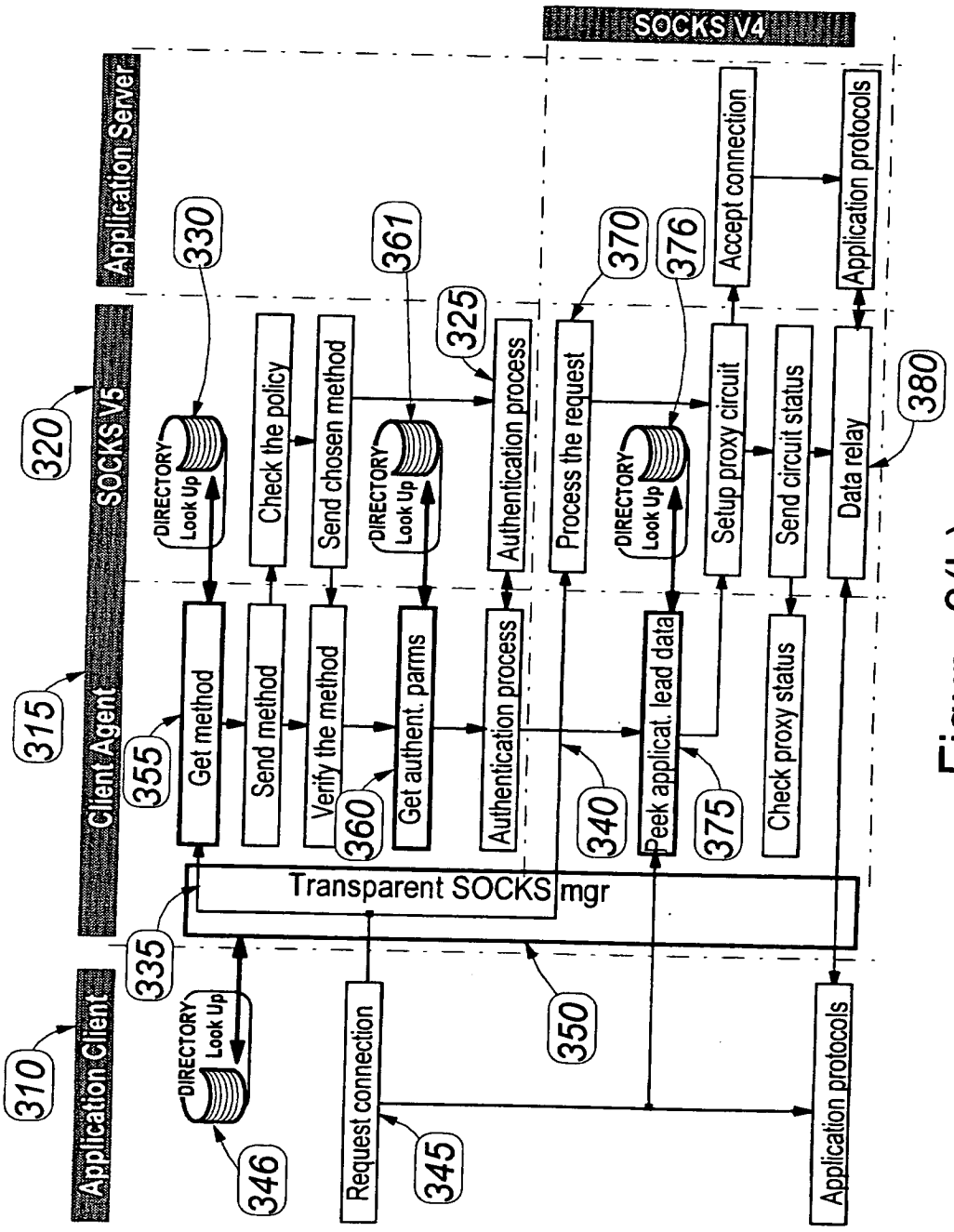

Figure 3(b)

FR 9 99 061